Adv. Appl. Clifford Algebras (2024) 34:6
© The Author(s), under exclusive licence to Springer Nature Switzerland AG 2024
0188-7009/010001-22
published online January 9, 2024
https://doi.org/10.1007/s00006-023-01308-5

Advances in Applied Clifford Algebras

Algorithms for Conic Fitting Through Given Proper and Improper Waypoints in Geometric Algebra for Conics

Pavel Loučka and Petr Vašík*

Abstract. As an addition to proper points of the real plane, we introduce a representation of improper points, i.e. points at infinity, in terms of Geometric Algebra for Conics (GAC) and offer possible use of both types of points. More precisely, we present two algorithms fitting a conic to a dataset with a certain number of points lying on the conic precisely, referred to as the waypoints. Furthermore, we consider inclusion of one or two improper waypoints, which leads to the asymptotic directions of the fitted conic. The number of used waypoints may be up to four and we classify all the cases.

Mathematics Subject Classification. Primary 15A66, Secondary 51N25.

Keywords. Conic fitting, Geometric algebra, Clifford algebra, Waypoint, Proper point, Improper point, Point at infinity, Ideal point.

Abbreviations

GAC	Geometric algebra for conics
CRA	Compass and ruler algebra
IPNS	Inner product null space

1. Introduction

Geometric Algebra for Conics (GAC), originally introduced in [12], and consequently elaborated in [5], is already acknowledged to be useful for conic manipulation, e.g. for intersections [1], and for simple conic fitting [6], as well

This article is part of the Topical Collection for the First International Conference on Advanced Computational Applications of Geometric Algebra (ICACGA) held in Denver, Colorado, USA, 2–5 October 2022, edited by David DaSilva, Eckhard Hitzer and Dietmar Hildenbrand.

*Corresponding author.

as conic fitting with additional geometric constraints such as axial alignment [9,10].

In the current paper, we present two conic fitting algorithms with various numbers of waypoints, meaning that besides a given dataset to be fitted we prescribe a set of points that should lie on the conic in question precisely. Both algorithms are modifications of the simple conic fitting algorithm using GAC as formulated in [6]. Let us also note that the first of the two algorithms for conic fitting through given waypoints in GAC has already been presented in [8], while the latter is new and removes some of the shortcomings of the first one. In particular, the first algorithm for conic fitting through waypoints does not necessarily fit the conic through all the given waypoints (even in geometrically meaningful cases) while the second algorithm does.

Additionally, we make a distinction between *proper points*, i.e. the points of the real plane $\mathbb{R}^2$, and *improper points*, i.e. the *points at infinity* (also called *ideal points*), and offer their representations in GAC. Consequently, we use both types of points as the waypoints for the conic fitting problem in GAC, thus allowing us to prescribe asymptotic direction(s) of the fitted conic in case of using the improper waypoint(s).

All our considerations follow from the description of conics in GAC, yet they can be used in the standard algorithms, such as Fitzgibbon's [3], too. On the other hand, we found only one reference to an algorithm imposing similar conditions on the fitted conic, more precisely, the conic centre was supposed to lie on a certain line [14]. The reason why GAC is suitable for designing similar algorithms is its geometric nature. Especially the improper waypoints are easily handled in the same way as proper points, i.e. as null vectors in Clifford algebra $\mathcal{C}l(5,3)$.

We note that the advantage of a GAC-oriented algorithm is again in the property that geometric objects are understood as elements of a Clifford algebra, i.e. vectors or multivectors, as well as their Euclidean transformations which are represented by bivectors. Object manipulation is then performed by sandwich product, i.e. geometric product intrinsic to GAC. Indeed, our motivation for conic fitting in GAC lies in the consequent simplicity of conic transformations, as shown, e.g. in [2] for the control of specific dynamical systems. On the other hand, actual Clifford algebra implementations do not provide great computational improvement, yet there exist some tools that promise computational acceleration in the future [4].

2. Proper and Improper Points in GAC

To clarify the significance of improper points, let us briefly recall the term *real projective plane* $\mathbb{RP}^2$ [13]:

Let $\mathbb{E} = (\mathcal{P}_\mathbb{E}, \mathcal{L}_\mathbb{E}, \mathcal{I}_\mathbb{E})$ be a usual Euclidean plane with points $\mathcal{P}_\mathbb{E} = \mathbb{R}^2$, lines $\mathcal{L}_\mathbb{E}$, and the usual incidence relation $\mathcal{I}_\mathbb{E} \subseteq \mathcal{P}_\mathbb{E} \times \mathcal{L}_\mathbb{E}$ of the Euclidean plane. Subsequently, by including elements at infinity, the Euclidean plane can be extended to the real projective plane.

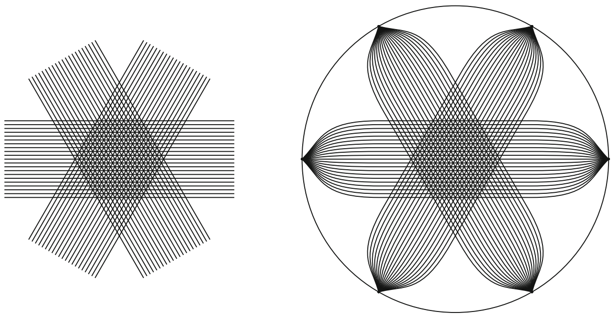


FIGURE 1. Parallel lines and their common improper points in $\mathbb{RP}^2$ (taken from [13])

First, let us assume line l and the equivalence class $[l]$ of all the lines parallel to l. An *improper point* $p_{[l]}$ is then defined as a common point of all the parallels contained in the equivalence class $[l]$, i.e. a point where all the parallels of the class meet. Moreover, all the improper points of $\mathbb{RP}^2$ form a line at infinity l_∞. Therefore, the definition of the *real projective plane* is as follows:

Definition 2.1. *Real projective plane* $\mathbb{RP}^2$ is a triple $(\mathcal{P}, \mathcal{L}, \mathcal{I})$, where

- $\mathcal{P} = \mathcal{P}_{\mathbb{E}} \cup \big\{p_{[l]} : l \in \mathcal{L}_{\mathbb{E}}\big\}$,
- $\mathcal{L} = \mathcal{L}_{\mathbb{E}} \cup l_\infty$,
- $\mathcal{I} = \mathcal{I}_{\mathbb{E}} \cup \big\{(p_{[l]}, l) : l \in \mathcal{L}_{\mathbb{E}}\big\} \cup \big\{(p_{[l]}, l_\infty) : l \in \mathcal{L}_{\mathbb{E}}\big\}$.

An example of three distinct bundles (equivalence classes) of parallel lines is illustrated in Fig. 1—on the left, the lines are depicted in plane $\mathbb{R}^2$; whereas on the right, we can see the situation in $\mathbb{RP}^2$: all the parallels from a particular bundle intersect at one common improper point situated on the line at infinity l_∞ (this line is drawn as a circle, since it can be thought of as a circle with an infinite radius). Moreover, despite the first sight, every bundle has exactly one improper point where all the parallels meet, not two; the two common antipodal points of each bundle depicted in Fig. 1 (right) are, in fact, geometrically coincident.

Unlike the proper points, the improper points are not elements of the plane $\mathbb{R}^2$, so an improper point is mostly depicted as a direction (vector) of the associated bundle of parallels, as shown in Fig. 2. Let us also note that sometimes the direction arrow points both ways to emphasise that the corresponding parallel lines stretch to both sides and their common improper point is—in a sense—infinitely distant on both sides, even though there is still only one.

Using the concept of improper points, it is easy to see that some types of conics pass through one or two improper points. In particular, a parabola has one improper point corresponding to the direction of its axis of symmetry, while a hyperbola has two improper points in the directions of its asymptotes, as illustrated in Fig. 3.

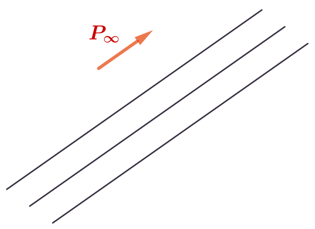


Figure 2. Depiction of an improper point P_∞ as a direction of parallel lines

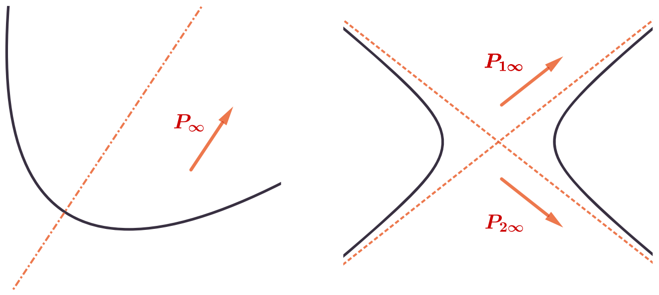


Figure 3. Improper points of parabola and hyperbola

2.1. Homogeneous Coordinates

Not only does the projective plane $\mathbb{RP}^2$ enable us to discern proper and improper points, it also allows us to perform computations with both types of points as if proper and improper points did not differ at all. Employing *homogeneous coordinates*, each point of $\mathbb{RP}^2$ (proper or improper) can be represented as a line in $\mathbb{R}^3$ in accordance with the following definition.

Definition 2.2. Let $\mathbf{x} = (x, y), \mathbf{x} \in \mathbb{RP}^2$, be a proper point, then its homogeneous coordinates are

$$\mathbf{x} = k(x, y, 1), \qquad k \in \mathbb{R} \setminus \{0\},$$

while the homogeneous coordinates of an improper point $\mathbf{x}_\infty = (s, t), \mathbf{x}_\infty \in \mathbb{RP}^2$, are

$$\mathbf{x}_\infty = k(s, t, 0), \qquad k \in \mathbb{R} \setminus \{0\}.$$

Remark 2.3. The triple $(0, 0, 0)$ does not represent any point of $\mathbb{RP}^2$.

Remark 2.4. As stated in Definition 2.2, homogeneous coordinates (a, b, c) represent the same point in $\mathbb{RP}^2$ as the triple $k(a, b, c)$ for every non-zero k. On the other hand, when one wants to assign homogeneous coordinates to the points of $\mathbb{RP}^2$, it is convenient to set $k = 1$. Consequently, to facilitate the future computations, we can consider the homogeneous coordinates of a proper point $\mathbf{x} = (x, y)$ and of an improper point $\mathbf{x}_\infty = (s, t)$, respectively, to be assigned by the mapping

$$\begin{aligned}(x,y) &\mapsto (x,y,1),\\ (s,t) &\mapsto (s,t,0).\end{aligned} \tag{2.1}$$

2.2. Projectivisation of GAC

Let us recall that GAC comprises a Clifford algebra $\mathcal{C}l(5,3)$ with an embedding $C:\mathbb{R}^2 \to \mathbb{R}^{5,3}$ of a proper point $\mathbf{x} = xe_1 + ye_2$ from the plane $\mathbb{R}^2$ to a six-dimensional subspace of one-vectors in GAC, in the form

$$C(x,y) = \bar{n}_+ + xe_1 + ye_2 + \frac{1}{2}(x^2+y^2)n_+ + \frac{1}{2}(x^2-y^2)n_- + xyn_\times \tag{2.2}$$

where $\{\bar{n}_\times, \bar{n}_-, \bar{n}_+, e_1, e_2, n_+, n_-, n_\times\}$ is the eight-dimensional vector basis of $\mathcal{C}\updownarrow(5,3)$ [6], together with an associated bilinear form of the inner product of vectors in GAC given by the matrix

$$B = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1\\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0\\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0\\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0\\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{2.3}$$

Note that the meaning of the basis vectors is standard as in any geometric algebra, i.e. $\bar{n}$'s denote the origins, e_1 and e_2 denote the basis vectors of the Euclidean plane and n's stand for infinities, respectively. Also note that the central 4×4 submatrix of the matrix B, i.e.

$$B_c = \begin{pmatrix} 0 & 0 & 0 & -1\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ -1 & 0 & 0 & 0 \end{pmatrix} \tag{2.4}$$

is the one corresponding to the subalgebra CRA. Consequently, the inner product null space (IPNS) representation of a general conic section Q in GAC is given by

$$Q_I = \bar{v}^\times \bar{n}_\times + \bar{v}^- \bar{n}_- + \bar{v}^+ \bar{n}_+ + v^1 e_1 + v^2 e_2 + v^+ n_+. \tag{2.5}$$

Equations of particular conics in GAC can be found in [1,5]. It is also well known that the type and features of conic Q can be read off its matrix representation [7], which is obtained using (2.5) as a matrix

$$M = \begin{pmatrix} -\frac{1}{2}(\bar{v}^+ + \bar{v}^-) & -\frac{1}{2}\bar{v}^\times & \frac{1}{2}v^1\\ -\frac{1}{2}\bar{v}^\times & -\frac{1}{2}(\bar{v}^+ - \bar{v}^-) & \frac{1}{2}v^2\\ \frac{1}{2}v^1 & \frac{1}{2}v^2 & -v^+ \end{pmatrix}.$$

Let us also note that a proper point embedded into GAC using mapping $C(x,y)$ of form (2.2) can be represented as a vector

$$P_I = \begin{pmatrix} 0 & 0 & 1 & x & y & \frac{1}{2}(x^2+y^2) & \frac{1}{2}(x^2-y^2) & xy \end{pmatrix}^T \tag{2.6}$$

and the IPNS conic section (2.5) as a vector

$$Q_I = \left(\bar{v}^\times\ \bar{v}^-\ \bar{v}^+\ v^1\ v^2\ v^+\ 0\ 0\right)^T. \tag{2.7}$$

Moreover, using the notion of homogeneous coordinates, we will show that GAC can be used to represent both proper and improper points. Namely, this GAC representation can be reached by extending the domain of the point embedding $C : \mathbb{R}^2 \to \mathbb{R}^{5,3}$ to $\mathbb{RP}^2$ according to the following definition.

Definition 2.5. Using the embedding $C : \mathbb{R}^2 \to \mathbb{R}^{5,3}$ of the form (2.2), we define the projective embedding $C\mathbb{P} : \mathbb{RP}^2 \to \mathbb{R}^{5,3}$ of a point $\mathbf{p} = (a,b,c)$, $(a,b,c) \neq (0,0,0)$, of the real projective plane $\mathbb{RP}^2$ as

$$C\mathbb{P}(a,b,c) = c^2\bar{n}_+ + ace_1 + bce_2 + \frac{1}{2}(a^2+b^2)n_+ + \frac{1}{2}(a^2-b^2)n_- + abn_\times. \tag{2.8}$$

Theorem 2.6. *Projective embedding $C\mathbb{P}$ maps proper and improper points of $\mathbb{RP}^2$, as described with homogenous coordinates in Sect. 2.1, to GAC.*

Proof. Let us recall the equation of a conic section in both $\mathbb{R}^2$ and $\mathbb{RP}^2$, respectively:

$$Q_{\mathbb{R}^2}: \quad q_{11}x^2 + 2q_{12}xy + q_2y^2 + 2q_{13}x + 2q_{23}y + q_{33} = 0, \tag{2.9}$$

$$Q_{\mathbb{RP}^2}: q_{11}x^2 + 2q_{12}xy + q_2y^2 + 2q_{13}xz + 2q_{23}yz + q_{33}z^2 = 0. \tag{2.10}$$

Next, let us remember that a conic Q_I in GAC is the IPNS representation of a conic $Q_{\mathbb{R}^2}$ if and only if $Q_{\mathbb{R}^2} = \left\{\mathbf{x} \in \mathbb{R}^2 : C(\mathbf{x}) \cdot Q_I = 0\right\}$, where $\cdot$ denotes the inner product between vectors in GAC [5]. Consequently, evaluating the equation

$$C(\mathbf{x}) \cdot Q_I = 0$$

from the aforementioned definition yields

$$-v^+ + v^1x + v^2y - \frac{1}{2}\bar{v}^+(x^2+y^2) - \frac{1}{2}\bar{v}^-(x^2-y^2) - \bar{v}^\times xy = 0. \tag{2.11}$$

As shown in [5], there is a direct relationship between the coefficients q_{ij} of $Q_{\mathbb{R}^2}$ and the coefficients of Q_I of the form

$$\begin{aligned} \bar{v}^\times &= -2q_{12}, \\ \bar{v}^- &= -(q_{11} - q_{22}), \\ \bar{v}^- &= -(q_{11} + q_{22}), \\ v^1 &= 2q_{13}, \\ v^2 &= 2q_{23}, \\ \bar{v}^\times &= -q_{33}. \end{aligned} \tag{2.12}$$

Substituting (2.12) into (2.11) and reordering the result then gives us an alternative representation of a conic $Q_{\mathbb{R}^2}$:

$$\begin{aligned} Q_{\mathbb{R}^2}: \frac{1}{2}(q_{11}+q_{22})(x^2+y^2) + 2q_{12}xy + \frac{1}{2}(q_{11}-q_{22})(x^2-y^2) \\ + 2q_{13}x + 2q_{23}y + q_{33} = 0, \end{aligned}$$

which is, after simplification, the same as (2.9). It is apparent that a conic $Q_{\mathbb{RP}^2}$ of form (2.10) is simply created from $Q_{\mathbb{R}^2}$ of form (2.9) by multiplying some of the terms by the third coordinate of a point, i.e. z, or its square, z^2. Analogously to IPNS representation of $Q_{\mathbb{R}^2}$, we can say that Q_I is the IPNS representation of $Q_{\mathbb{RP}^2}$ if and only if $Q_{\mathbb{RP}^2} = \left\{\mathbf{x} \in \mathbb{RP}^2 : C\mathbb{P}(\mathbf{x}) \cdot Q_I = 0\right\}$. Therefore, the projective embedding $C\mathbb{P}$ must be defined in such a way that the equation

$$C\mathbb{P}(\mathbf{x}) \cdot Q_I = 0$$

is the same as (2.10). As in the case of the transition from $Q_{\mathbb{R}^2}$ to $Q_{\mathbb{RP}^2}$, embedding $C\mathbb{P}$ of the form (2.8) is attained by multiplying particular terms of the embedding C, (2.2), by the third coordinate of a point or its square. $\square$

Corollary 2.7. *Since a proper point $\mathbf{x} = (x, y)$ has homogeneous coordinates $(x, y, 1)$, both the projective embedding $C\mathbb{P}$ and the embedding C maps a proper point into GAC without difference:*

$$C\mathbb{P}(x, y, 1) \equiv C(x, y) = \bar{n}_+ + xe_1 + ye_2 + \frac{1}{2}(x^2 + y^2)n_+ + \frac{1}{2}(x^2 - y^2)n_- + xyn_\times.$$

Besides, an improper point $\mathbf{x}_\infty = (s, t)$ with homogeneous coordinates $(s, t, 0)$ is embedded into GAC in a similar way:

$$C\mathbb{P}(s, t, 0) = \frac{1}{2}(s^2 + t^2)n_+ + \frac{1}{2}(s^2 - t^2)n_- + stn_\times. \tag{2.13}$$

Subsequently, creating an analogy with the IPNS vector form (2.6) of a proper point, we can define an IPNS vector form of an improper point as

$$P_{\infty I} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & \frac{1}{2}(s^2 + t^2) & \frac{1}{2}(s^2 - t^2) & st \end{pmatrix}^T. \tag{2.14}$$

Remark 2.8. Let us also note that, in contrast with embedding of a proper point, an improper point embedded into GAC according to (2.13) is always expressed using the basis vectors corresponding to infinities of Witt pairs, only.

3. Conic Fitting in GAC—Fitting Without Waypoints

Let us briefly recall the usual conic fitting (i.e. fitting without given waypoints) in terms of GAC. Hrdina, Návrat and Vašík [6], define a conic fitting problem using GAC in the following way: For N_D data points embedded into GAC as vectors P_i of the form (2.6) and for a conic represented by a vector Q of the form (2.7), let us minimise the objective function

$$Q \mapsto \sum_i (P_i \cdot Q)^2, \tag{3.1}$$

i.e. a sum of squared point-to-conic algebraic distances. Since the conic vector $Q = 0$ is geometrically meaningless, the authors of [6] also assume the normalisation constraint

$$Q^2 = 1. \tag{3.2}$$

Applying the matrix of the bilinear form (2.3), the objective function (3.1) can be rewritten as

$$Q \mapsto \sum_i (P_i B Q)^2 = \sum_i Q^T B P_i P_i^T B Q = Q^T P Q, \tag{3.3}$$

and thus, it is a quadratic form on $\mathbb{R}^{5,3}$ with the matrix

$$P = \sum_i B P_i P_i^T B.$$

Moreover, the normalisation constraint (3.2) is equivalently expressed as

$$Q^T B Q = 1. \tag{3.4}$$

Consequently, conic fitting in GAC can be treated as an optimisation problem and formulated using vector–matrix notation as

$$\begin{aligned} \min_Q \quad & Q^T P Q \\ \text{s.t.} \quad & Q^T B Q = 1. \end{aligned} \tag{3.5}$$

To formulate the solution to this problem, the authors decided to decompose matrix P into matrix blocks as follows:

$$P = \begin{pmatrix} P_0 & P_1 & 0_{2\times 2} \\ P_1^T & P_c & 0_{4\times 2} \\ 0_{2\times 2} & 0_{2\times 4} & 0_{2\times 2} \end{pmatrix}. \tag{3.6}$$

As in matrix B_c of the form (2.4), the "c" subscript in matrix P_c signifies a central 4×4 part of matrix P, i.e. a corresponding CRA part of GCA.

Finally, using the vectors and matrices defined before, the solution to the conic fitting problem is obtained in the following way.

Proposition 3.1. *The solution to the optimisation problem (3.5) for conic fitting in GAC is given by* $Q = \begin{pmatrix} w^T & v^T & 0 & 0 \end{pmatrix}^T$, *where* $v = \begin{pmatrix} \bar{v}^+ & v^1 & v^2 & v^+ \end{pmatrix}^T$ *is an eigenvector corresponding to the minimal non-negative eigenvalue of the operator*

$$P_{con} = B_c (P_c - P_1^T P_0^{-1} P_1)$$

and $w = \begin{pmatrix} \bar{v}^\times & \bar{v}^- \end{pmatrix}^T$ *is a vector acquired as*

$$w = -P_0^{-1} P_1 v.$$

The proof of Proposition 3.1 and the corresponding algorithm implemented in MATLAB together with the experimental results can be found in [6,9,10].

4. Conic Fitting in GAC—Fitting with Given Waypoints

To fit a conic among the data points as closely as possible and to simultaneously guarantee its passage through given waypoint(s), we can make use of the vector–matrix formulation of conic fitting in terms of GAC, as described in Sect. 3. Indeed, the conic fitting problem without prescribed waypoints and with them are similar in structure; thus, as will be shown further, both types of problems can be solved using a certain eigenproblem.

Following the example of conic fitting without waypoints, we consider N_D data points represented by vectors P_i of the form (2.6) and we want to obtain a conic vector Q of the form (2.7) that minimises the objective function (3.1) and satisfies the normalisation constraint (3.2). Let us also stress that *proper* points only will be further used as the data points P_i, as fitting a conic among the improper points and not through them is geometrically unjustified.

Additionally, we require the conic Q to pass through N_W waypoints W_j, which can be—in contrast with data points P_i—both proper and improper. Hence, each of them is either of the form (2.6) or (2.14). A conic Q passes through waypoint W if and only if their inner product vanishes, i.e.

$$W \cdot Q = 0.$$

After defining the matrix Ψ of waypoints W_j, where the j-th column is waypoint W_j, the condition of conic Q passing through all the waypoints W_j can be expressed as

$$\Psi \cdot Q = 0,$$

or, equivalently, using vectors and matrices, as

$$\Psi^T BQ = 0. \tag{4.1}$$

Consequently, the examined conic fitting problem takes the form

$$\begin{aligned} \min_{Q} \quad & Q^T PQ \\ \text{s.t.} \quad & Q^T BQ = 1, \\ & \Psi^T BQ = 0. \end{aligned} \tag{4.2}$$

In the following subsection, let us describe two conic fitting algorithms solving the given optimisation problem.

4.1. Algorithms

4.1.1. Algorithm QW-pseudoinv. First, let us recapitulate and further describe Algorithm **QW-pseudoinv**, which has already been introduced in [8] as Algorithm **QW**.[1]

To express the solution to the given optimisation problem, let us define the matrix

$$B_0 = \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix},$$

[1]The renaming of the algorithm was done in order to distinguish it from a new algorithm which fits a conic through given waypoint(s) as well. Moreover, the new name of the algorithm stresses that it reaches the solution using the Moore–Penrose pseudoinverse.

and decompose the matrix Ψ of waypoints W_j as

$$\Psi = \left(\begin{array}{c|c|c|c} W_1 & W_2 & \cdots & W_{N_W} \end{array} \right) = \left(\begin{array}{c} 0 \\ \hline \Xi \\ \hline X \end{array} \right),$$

where 0 denotes a $2 \times N_W$ zero matrix, and Ξ, X stand for matrices of types $4 \times N_W$ and $2 \times N_W$, respectively.

Next, we consider the matrix

$$B_w = \left(B_0^T X\right)^{+^T} \Xi^T B_c, \tag{4.3}$$

where "+" signifies the Moore–Penrose pseudoinverse, as the matrix $B_0^T X$ is necessarily not square.

Similarly to Proposition 3.1, Algorithm **QW-pseudoinv** reaches a solution to the conic fitting problem (4.2) using an eigenproblem given in the following statement.

Proposition 4.1. *The solution to the optimisation problem* (4.2) *for conic fitting in GAC is given by* $Q = \left(w^T \ v^T \ 0 \ 0\right)^T$, *where* $v = \left(\bar{v}^+ \ v^1 \ v^2 \ v^+\right)^T$ *is an eigenvector corresponding to the minimal non-negative eigenvalue of the operator*

$$P_{con}^W = B_c \left[B_w^T P_0 B_w - \left(B_w^T P_1 + P_1^T B_w\right) + P_c\right] \tag{4.4}$$

and $w = \left(\bar{v}^\times \ \bar{v}^-\right)^T$ *is a vector acquired as*

$$w = -B_w v.$$

Proof. Using the method of Lagrange multipliers and applying it on problem (4.2), we would obtain a Lagrange function with two multipliers and the corresponding system of linear equations that, unfortunately, would be difficult to solve.

To reduce the complexity of the problem, we chose to incorporate the waypoints constraint (4.1) into both the objective function (3.3) and the normalisation constraint (3.4). We can see that

$$\Psi^T B Q = 0 \iff \Xi^T B_c v + X^T B_0 w = 0,$$

thus

$$X^T B_0 w = -\Xi^T B_c v, \tag{4.5}$$

and, using matrix B_w defined in (4.3), we get

$$w \approx -B_w v$$

Next, if the waypoints constraint is omitted in the optimisation problem and only the objective function and the normalisation constraint are assumed, then the corresponding Lagrange function has one multiplier only and takes the form

$$P_\lambda(Q) = Q^T P Q + \lambda(1 - Q^T B Q).$$

Using the vectors w and v, the acquired Lagrange function can be further expanded as

$$P_\lambda(w, v) = w^T P_0 w + 2w^T P_1 v + v^T P_c v + \lambda(1 - v^T B_c v).$$

After the substitution of $-B_w v$ for w, we obtain

$$P_\lambda(v) = v^T B_w^T P_0 B_w v - 2v^T B_w^T P_1 v + v^T P_c v + \lambda(1 - v^T B_c v).$$

Differentiation of this Lagrange function with respect to v and λ results in the system of linear equations

$$\begin{aligned} \frac{\partial P_\lambda}{\partial v} &= 2\left[\left(B_w^T P_0 B_w - \left(B_w^T P_1 + P_1^T B_w\right) + P_c\right) v - \lambda B_c v\right] = 0, \\ \frac{\partial P_\lambda}{\partial \lambda} &= 1 - v^T B_c v = 0. \end{aligned} \tag{4.6}$$

The first of the two equations further implies that

$$\left[B_w^T P_0 B_w - \left(B_w^T P_1 + P_1^T B_w\right) + P_c\right] v = \lambda B_c v.$$

Since matrix B_c squares to identity, multiplying the equation by this matrix from left yields

$$B_c \left[B_w^T P_0 B_w - \left(B_w^T P_1 + P_1^T B_w\right) + P_c\right] v = \lambda v,$$

which constitutes a standard eigenproblem for v and λ. Furthermore, v must be normalised according to the second equation in (4.6), i.e.

$$v^T B_c v = 1. \tag{4.7}$$

Therefore, the v-part of each stationary point of the Lagrange function is an eigenvector of the matrix (4.4). Unfortunately, this is not always possible because B_c is not positive definite. To find the minimum among the stationary points, let us compute the values of objective function (3.1): Let v_λ be the eigenvector corresponding to eigenvalue λ and Q_λ be the corresponding conic $Q_\lambda = (-B_w v_\lambda \quad v_\lambda \quad 0)^T$. Using the blocks of matrix P, the system of linear equations 4.6 and the normalisation constraint (4.7), we obtain

$$\begin{aligned} P_\lambda(Q_\lambda) = Q_\lambda^T P Q_\lambda = v^T \left[B_w^T P_0 B_w - \left(B_w^T P_1 + P_1^T B_w\right) + P_c\right] v = \lambda v^T B_c v = \\ = \lambda > 0. \end{aligned}$$

Consequently, the eigenvectors corresponding to non-positive eigenvalues cannot be normalised as in (4.7) and are not stationary points. Moreover, the minimum of the objective function (3.3) is achieved if v is the eigenvector associated with the least positive eigenvalue of operator P_{con}^W. $\square$

4.1.2. Algorithm QW-null. To also cover the cases in which Algorithm **QW-pseudoinv** fails to fit the conic through all the waypoints (as will be further elaborated later) we introduce a new conic fitting algorithm, Algorithm **QW-null**. As indicated by its name, the algorithm computes the fitted conic passing through given waypoint(s) using a null space of a certain matrix.

As in the case of the previous algorithm, we need to define a few auxiliary vectors and matrices first. It can be seen that some of the vectors and matrices have elements that equal zero by definition—in some cases, though, we can omit these elements since they may result in unnecessary computational operations, and, moreover, omitting them may also greatly help in expressing the solution to the optimisation problem. For example, by omitting the two last zero elements in the IPNS vector representation of a conic Q of the form (2.7), we can define a *reduced* IPNS conic vector as

$$\hat{Q} = \begin{pmatrix} \bar{v}^{\times} & \bar{v}^{-} & \bar{v}^{+} & v^1 & v^2 & v^{+} \end{pmatrix}^T .$$

Analogously, we can omit the first two zero elements in the IPNS vector of a proper point P of the form (2.6) and an improper point P_∞ of the form (2.14) and create reduced points

$$\begin{aligned} \hat{P}_I &= \begin{pmatrix} 1 & x & y & \frac{1}{2}(x^2+y^2) & \frac{1}{2}(x^2-y^2) & xy \end{pmatrix}^T , \\ \hat{P}_{\infty I} &= \begin{pmatrix} 0 & 0 & 0 & \frac{1}{2}(x^2+y^2) & \frac{1}{2}(x^2-y^2) & xy \end{pmatrix}^T . \end{aligned}$$

Thus, the matrix of reduced waypoints reads

$$\hat{\Psi} = \left(\begin{array}{c|c|c|c} \hat{W}_1 & \hat{W}_2 & \cdots & \hat{W}_{N_W} \end{array} \right) = \left(\begin{array}{c} \Xi \\ \hline X \end{array} \right) .$$

In a similar way, we can omit the two last rows and columns in matrix P of the form (3.6) and create a reduced matrix

$$\hat{P} = \begin{pmatrix} P_0 & P_1 \\ P_1^T & P_c \end{pmatrix} .$$

Furthermore, we define two matrices reduced from the matrix of bilinear form by omitting two rows and columns of matrix B of the form (2.3), in particular we have

$$\hat{B}_1 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \end{pmatrix}, \quad \hat{B}_2 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} .$$

Consequently, Algorithm **QW-null** reaches the solution as follows.

Proposition 4.2. *The solution to the optimisation problem* (4.2) *for conic fitting in GAC is given by* $Q = \begin{pmatrix} \hat{Q}^T & 0 & 0 \end{pmatrix}^T$*, with* $\hat{Q} = \begin{pmatrix} \bar{v}^{\times} & \bar{v}^{-} & \bar{v}^{+} & v^1 & v^2 & v^{+} \end{pmatrix}^T$ *computed as*

$$\hat{Q} = Nc,$$

where

$$N = \operatorname{null}\left(\hat{\Psi}^T \hat{B}_1 \right),$$

i.e. a matrix of the orthonormal basis of the null space of matrix $\hat{\Psi}^T \hat{B}_1$*, and* c *is an eigenvector corresponding to the least positive eigenvalue of the*

Table 1. Combinations of waypoints by type and total number

	1A	1B	2A	2B	2C	3A	3B	3C	3D	4A	4B	4C	4D	4E
Proper	1	0	2	1	0	3	2	1	0	4	3	2	1	0
Improper	0	1	0	1	2	0	1	2	3	0	1	2	3	4

generalised eigenproblem

$$N^T\hat{P}Nc = \lambda N^T\hat{B}_2Nc.$$

Proof. For conic Q to pass through all the waypoints given, the constraint (4.1) must be fulfilled. Moreover, we can express this condition equivalently as follows:

$$\Psi^T BQ = 0 \iff \hat{\Psi}^T\hat{B}_1\hat{Q} = 0.$$

Therefore, vector $\hat{Q}$ must lie in the null space of matrix $\hat{\Psi}^T\hat{B}_1$. This implies that $\hat{Q}$ must be a linear combination of the columns of matrix N, or, in vector–matrix notation

$$\hat{Q} = Nc.$$

Thus,

$$\Psi^T BQ = 0 \iff \hat{Q} = Nc.$$

After constructing the corresponding Lagrange function and substituting $\hat{Q} = Nc$ to it, the rest of the proof proceeds in a way analogous to Proposition 4.1. □

4.2. Implementation

We offer MATLAB implementation of the two presented conic fitting algorithms with waypoints in section A of the Appendix.

4.3. Number of Waypoints and Degrees of Freedom of the Fitted Conic

As discussed in detail in [11], a conic is generally determined by five points of $\mathbb{RP}^2$. Hence, it is not possible to fit a conic through more than four waypoints and among the data points at the same time. Therefore, a maximum of four waypoints comes into play. Since we can use either proper or improper waypoints or even a combination of both types of waypoints, we get 14 cases of what waypoints to employ in total, as can be seen in Table 1.

Nevertheless, we must also take into account that some of the combinations given by Table 1 are geometrically unreasonable. In particular, no conic can have more than two improper waypoints. Therefore, cases 3D, 4D and 4E will not be considered further.

4.4. Experimental Results

We applied both of the presented conic fitting algorithms with given waypoint(s) to three different datasets listed in Table 2 (the location of the points in each dataset was deliberately chosen to resemble the shape of a regular

Table 2. Datasets used

	Elliptical									Parabolic								
x_i	3	4	3	0	−1	−3	−4	−3	−1	−2	−2	−1	−1	0	1	3	5	7
y_i	−1	1	2	3	3	2	−1	−3	−4	0	3	−2	6	8	−2	−2	−1	0
	Hyperbolic																	
x_i	−6	−4	−4	−3	−3	1	2	2	4									
y_i	1	2	−3	1	−1	5	4	7	4									

Table 3. Waypoints used and the corresponding cases

	1-4A				1B	2B		2C		3B		3C				4B				4C			
x_j	2	3	−4	−2	4	−7	1	6	1	−7	1	6	5	1	6	−7	6	1	6	−7	6	1	6
y_j	−3	3	1	−4	5	1	2	1	2	1	4	1	4	2	1	1	4	4	1	1	4	2	1
z_j	1	1	1	1	0	1	0	0	0	1	1	0	1	0	0	1	1	1	0	1	1	0	0

conic; hence, the corresponding datasets were named *elliptical*, *parabolic* and *hyperbolic*, respectively).

Moreover, by varying the total number and types of the waypoints used, we offer all 11 plausible cases of fits from Table 1. The particular waypoints used in the experiments are listed in Table 3.

In Fig. 4 we can see four cases of fitting the elliptical dataset, each fit being the result of fitting a conic with one more proper waypoint than in the preceding subfigure (see Table 3). In case 1A, both algorithms fit an ellipse passing through the waypoint, as expected. Although both algorithms work in case 2A as well, the result is still somehow unexpected, since both fitted conics are not only similar, they are actually the same as will be further discussed below.

A striking difference between the two algorithms can be seen in cases 3A and 4A—while **QW-pseudoinv** fails to fit a conic through the given waypoints, **QW-null** succeeds in that. This behaviour of the first-mentioned algorithm is caused by using the Moore–Penrose pseudoinverse in the auxiliary computation of matrix B_w, (4.3), which generally cannot guarantee the fulfilment of the waypoints constraint (4.1). In particular, when we employ more than two waypoints in our conic fitting problem, the LHS of Eq. (4.5) generally represents the LHS of an overdetermined system of linear equations in terms of the elements of vector w, and, thus, Eq. (4.5) does not necessarily hold. On the other hand, conics fitted by **QW-null** pass (and must pass) through the given waypoints because of the definition of the solution given as a part of the null space of matrix $\hat{\Psi}^T \hat{B}_1$.

Figure 5 depicts a situation of fitting a conic through one improper waypoint, and, although not explicitly, it shows another significant difference between the two algorithms—they fitted two different types of conics. It can

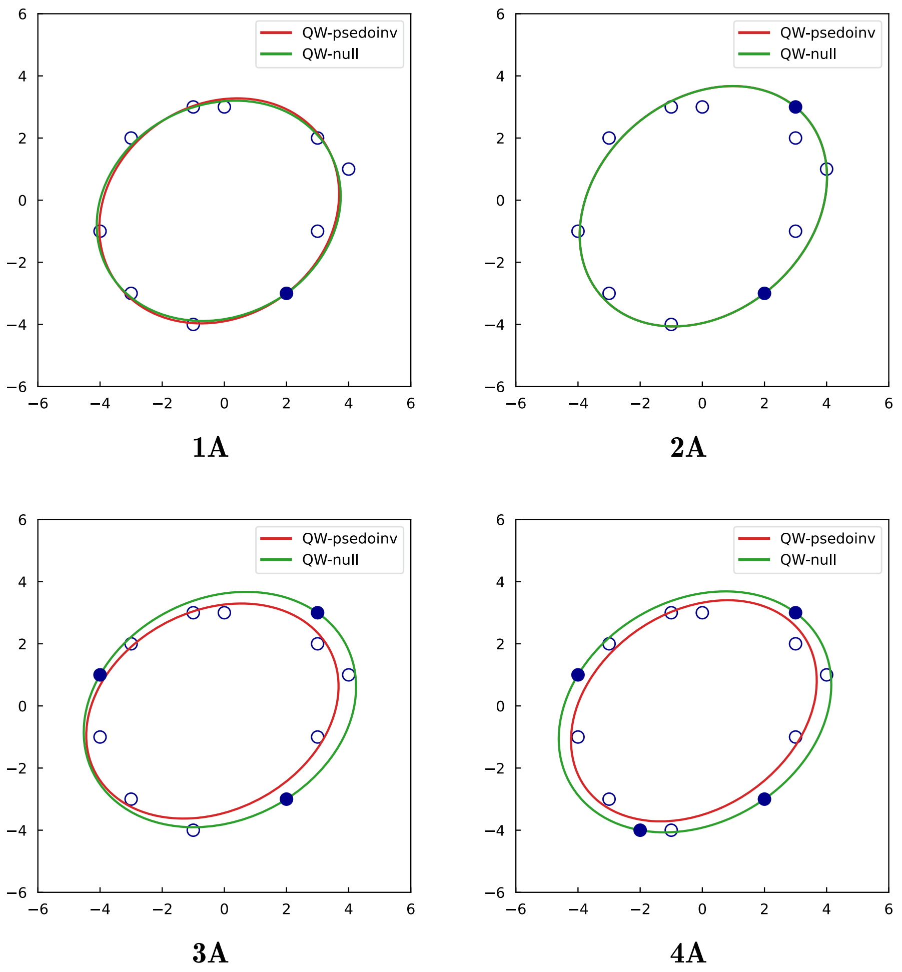
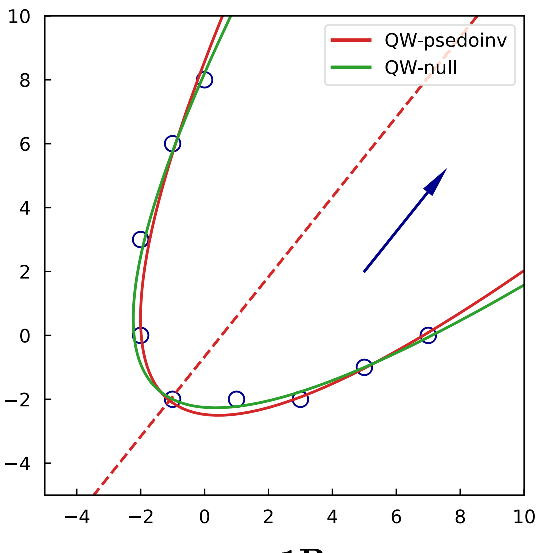


Figure 4. Conic fits through 1–4 proper waypoints

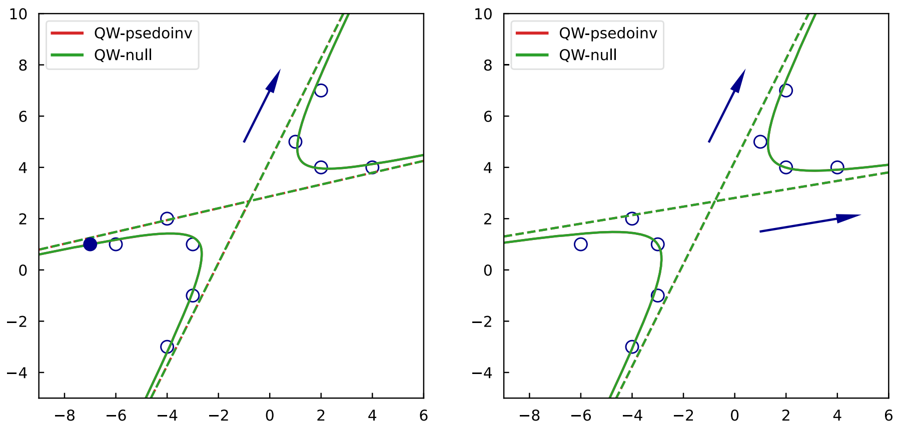
be proven that in case 1B Algorithm **QW-pseudoinv** generally fits a parabola with an axis of symmetry pointing in the direction of the improper waypoint, while **QW-null** fits a hyperbola with one asymptote passing through the same waypoint (the mentioned asymptote lies outside the figure). In contrast with a hyperbolic fit, obtaining a parabolic fit is, in general, geometrically unusual, and in the case of **QW-pseudoinv** it results from the fact that most elements in matrix B_w are zero when exactly one improper waypoint is used.

Fits through two waypoints are offered in Fig. 6, namely, 2B shows a fit through one proper and one improper waypoint and 2C through two improper waypoints. As in case 2A, both algorithms in both cases produce equivalent, visually overlapping results, and the fitted conics truly pass through their respective waypoints. Moreover, we can state that both algorithms are equivalent when two waypoints (whether proper or improper) are employed—for



1B

Figure 5. Fit through one improper waypoint



2B **2C**

Figure 6. Fits through two waypoints (without case 2A)

two waypoints the matrix $X^T B_0$ from Eq. (4.5) becomes square and the eigenproblems in the underlying Propositions 4.1 and 4.2 can be further proven equivalent as well.

Fits through three waypoints given in Fig. 7 are well in accordance with the result in case 3A; even though it is not entirely obvious, both a detailed examination of the figure and a computational verification confirm that, again, Algorithm **QW-pseudoinv** fails to fit the conics through the given

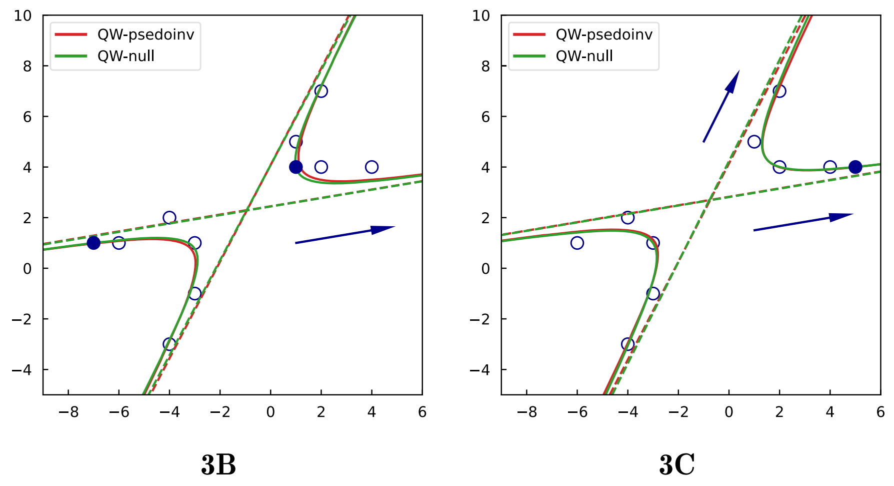


FIGURE 7. Fits through three waypoints (without case 3A)

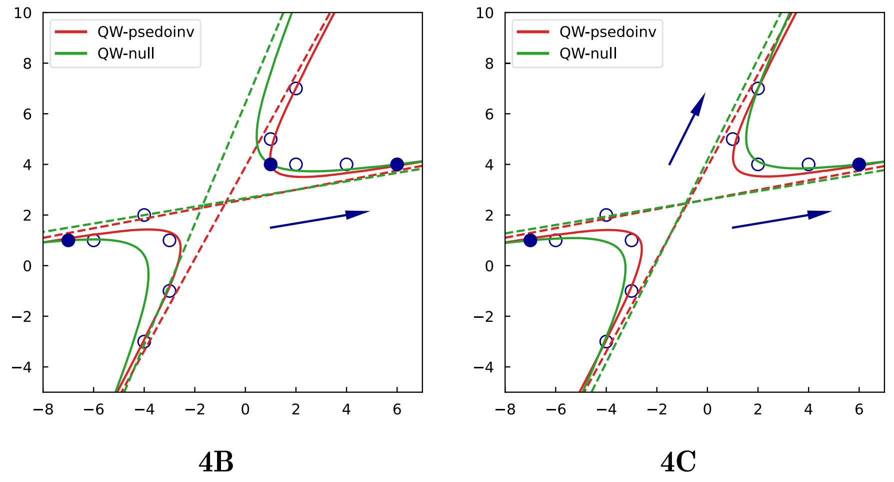


FIGURE 8. Fits through four waypoints (without case 4A)

waypoints (even though it approximates them quite closely) while the conics generated by **QW-null** indeed pass through the waypoints.

The remaining fits through the four waypoints shown in Fig. 8 only affirm—if not visually, then definitely by computation—that Algorithm **QW-pseudoinv** generally cannot fit a conic through a higher number of points. Nonetheless, it must be emphasised that while Algorithm **QW-null** was able to fit conics through all four waypoints, it came at a high price, namely, quite a high value of the objective function compared with **QW-pseudoinv**. This corresponds to the lower level of the fit's tightness among the data points, which is especially apparent in case 4B.

Table 4. Overview of the ability of algorithms to fit a conic through all the given waypoints in the respective cases (implausible cases are denoted by "–")

	1A	1B	2A	2B	2C	3A	3B	3C	3D	4A	4B	4C	4D	4E
QW-pseudoinv	✓	✓	✓	✓	✓	×	×	×	–	×	×	×	–	–
QW-null	✓	✓	✓	✓	✓	✓	✓	✓	–	✓	✓	✓	–	–

All in all, we can say that while Algorithm **QW-pseudoinv** is generally not capable of fitting a conic through more than two waypoints, Algorithm **QW-null** can, by definition, fit a conic passing through up to four waypoints. Also, let us reiterate that both algorithms produce equivalent conics when fitting through two waypoints. An overview of the success rates of both algorithms, i.e. their ability to fit a conic through all the given waypoints in the respective cases, is given by Table 4.

5. Conclusions

We presented two conic fitting algorithms with additional conditions in the form of a set of prescribed waypoints. This set may contain up to four points from which up to two can be improper. It turns out that the representation of improper points in GAC is natural, and thus all the intrinsic operations of GAC may be carried out in the same way as for proper points.

We recall that our two algorithms differ in the way how the optimisation problem is treated. One is based on the Moore–Penrose pseudoinverse and the other on search for the kernel of a linear operator. The latter algorithm proves itself more appropriate as it works even in the case of more than two waypoints where the first algorithm fails.

We modified the pseudoinverse-based fitting algorithm from [8] to the kernel-based one and provided their comparison. Note that we omitted a comparison to classical algorithms, see e.g. [6], which would have to be modified first to work with waypoints. The reason is that we do not expect anything different from the comparison of the standard algorithms in [6]. Also, the adaptation of the standard algorithms to our setting is possible but not of our interest. Indeed, a projectivisation of $\mathbb{R}^2$ must be considered leading to an additional coordinate and the algorithms would have to be modified accordingly.

Acknowledgements

The research was supported by a Grant no. FSI-S-23-8161.

Data Availability Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Appendix A. MATLAB Implementation of Conic Fitting Algorithms

Below, we summarise both presented algorithms for conic fitting with given waypoints implemented as a MATLAB function. Let us note that the reduced forms of some vectors and matrices were employed to avoid a few unnecessary computations with zero elements, similarly to the conic fitting algorithms in [9,10].

Both algorithms receive the same types of inputs and generate the same types of outputs, in particular:

Inputs:

a, **b**, **c**	column vectors of x, y, z homogeneous coordinates of waypoints
px, **py**	column vectors of x, y coordinates of data points

Outputs:

Conic	fitted conic in the form (2.7)
obj_function	value of objective function (3.1) for fitted conic

Let us also reiterate that when given affine coordinates of a proper point or the elements of a vector corresponding to an improper point, these can be easily converted into homogeneous coordinates using the mapping (2.1).

```
function [Conic, obj_function] = QW_pseudoinv(a,b,c,px,py)
ND = length(px);

Xi = [c a.*c b.*c 1/2*(a.^2+b.^2)]';
Chi = [1/2*(a.^2-b.^2) a.*b]';

B = zeros(6);
I3 = [0 0 1;0 1 0; 1 0 0];
B(1:3,4:6) = -I3;
B(4:5,2:3) = eye(2);
B(6,1) = -1;
Bc = B(3:6,1:4);
B0 = [0 -1;
-1 0];
Bw = (pinv(B0'*Chi))'*Xi'*Bc;

D = ones(6,ND);
D(2,:) = px;
D(3,:) = py;
D(4,:) = 1/2*(px.^2+py.^2);
D(5,:) = 1/2*(px.^2-py.^2);
D(6,:) = px.*py;

P = 1/ND*B*(D*D')*B';
Pc = P(3:6,3:6);
P0 = P(1:2,1:2);
P1 = P(1:2,3:6);

PWcon = Bc*(Bw'*P0*Bw-(Bw'*P1+P1'*Bw)+Pc);
[EV,ED] = eig(PWcon);
EW = diag(ED);

k_opt = find(EW == min(EW(EW>0)));
v_opt = EV(:,k_opt);

kappa = v_opt'*Bc*v_opt;
v_opt = 1/sqrt(kappa)*v_opt;

w = -Bw*v_opt;

Conic = [w;v_opt;0;0];
obj_function = Conic(1:6)'*P*Conic(1:6);
end
```

```
function [Conic, obj_function] = QW_null(a,b,c,px,py)
ND = length(px);
NW = length(a);

Psi = zeros(6,NW);
Psi(1,:) = c.^2;
Psi(2,:) = a.*c;
Psi(3,:) = b.*c;
Psi(4,:) = 1/2*(a.^2+b.^2);
Psi(5,:) = 1/2*(a.^2-b.^2);
Psi(6,:) = a.*b;

B = zeros(8);
I3 = [0 0 1;0 1 0; 1 0 0];
B(1:3,6:8) = -I3;
B(4:5,4:5) = eye(2);
B(6:8,1:3) = -I3;
B1 = B(3:8,1:6);
B2 = B(1:6,1:6);

N = null(Psi'*B1);

D = ones(6,ND);
D(2,:) = px;
D(3,:) = py;
D(4,:) = 1/2*(px.^2+py.^2);
D(5,:) = 1/2*(px.^2-py.^2);
D(6,:) = px.*py;

P = 1/ND*B1'*(D*D')*B1;

[EV,ED] = eig(N'*P*N,N'*B2*N);
EW = diag(ED);

k_opt = find(EW == min(EW(EW>0)));
c_opt = EV(:,k_opt);

Q_opt = N*c_opt;
kappa = Q_opt'*B2*Q_opt;
Q_opt = 1/sqrt(kappa)*Q_opt;

Conic = [Q_opt;0;0];
obj_function = Q_opt'*P*Q_opt;
end
```

References

[1] Byrtus, R., Derevianko, A., Vašík, P., Hildenbrand, C., Steinmetz, C.: On Specific Conic Intersections in GAC and Symbolic Calculations in GAALOPWeb. Advances in Applied Clifford Algebras, vol. 32, no. 1, ISSN 0188-7009 (2022). https://doi.org/10.1007/s00006-021-01182-z

[2] Derevianko, A.I., Vašík, P.: Solver-free optimal control for linear dynamical switched system by means of geometric algebra. Math. Methods Appl. Sci. (2022). https://doi.org/10.1002/mma.8752

[3] Fitzgibbon, A.W., Pilu, M., Fisher, R.B.: Direct least squares fitting of ellipses. IEEE Trans. Pattern Anal. Mach. Intell. **21**(5), 476–480 (1999)

[4] Hildenbrand, D.: Introduction to Geometric Algebra Computing. CRC Press, Taylor & Francis Group, Boca Raton (2019)

[5] Hrdina, J., Návrat, A., Vašík, P.: Geometric algebra for conics. Adv. Appl. Clifford Algebras **28**, 66 (2018). https://doi.org/10.1007/s00006-018-0879-2

[6] Hrdina, J., Návrat, A., Vašík, P.: Conic fitting in geometric algebra setting. Adv. Appl. Clifford Algebras **29**, 72 (2019). https://doi.org/10.1007/s00006-019-0989-5

[7] Korn, G.A., Korn, T.M.: Mathematical Handbook for Scientists and Engineers. McGraw-Hill Book Company, New York (1961)

[8] Loučka, P.: On Proper and Improper Points in Geometric Algebra for Conics and Conic Fitting Through Given Waypoints. ENGAGE,: Lecture Notes in Computer Science, vol. 13862. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-30923-6_6

[9] Loučka, P., Vašík, P.: Algorithms for Multi-conditioned Conic Fitting in Geometric Algebra for Conics. Advances in Computer Graphics. CGI 2021. Lecture Notes in Computer Science, vol. 13002. https://doi.org/10.1007/978-3-030-89029-2_48

[10] Loučka, P., Vašík, P.: On multi-conditioned conic fitting in geometric algebra for conics. Adv. Appl. Clifford Algebras **33**, 31 (2023). https://doi.org/10.1007/s00006-023-01277-9

[11] Pamfilos, P.: A gallery of conics by five elements. Forum Geom. **14**, 295–348 (2014)

[12] Perwass, C.: Geometric Algebra with Applications in Engineering. Springer, Berlin (2009)

[13] Richter-Gebert, J.: Perspectives on Projective Geometry: A Guided Tour Through Real and Complex Geometry. Springer, Berlin (2016)

[14] Waibel, P., Matthes, J., Gröll, L.: Constrained ellipse fitting with center on a line. J. Math. Imaging Vis. **53**, 364–382 (2015)

Pavel Loučka and Petr Vašík
Brno University of Technology
Technická 2896/2
616 69 Brno
Czech Republic
e-mail: `Petr.Vasik@vutbr.cz`

Pavel Loučka
e-mail: `Pavel.Loucka@vutbr.cz`

Received: February 17, 2023.
Accepted: October 30, 2023.